Method

Application of independent component analysis to microarrays

Su-In Lee* and Serafim Batzoglou†

Addresses: *Department of Electrical Engineering, Stanford University, Stanford, CA94305-9010, USA. †Department of Computer Science, Stanford University, Stanford, CA94305-9010, USA.

Correspondence: Serafim Batzoglou. E-mail: serafim@cs.stanford.edu

Published: 24 October 2003

Genome ***Biology*** 2003, **4:**R76

The electronic version of this article is the complete one and can be found online at http://genomebiology.com/2003/4/11/R76

Received: 10 March 2003
Revised: 27 June 2003
Accepted: 4 September 2003

Abstract

We apply linear and nonlinear independent component analysis (ICA) to project microarray data into statistically independent components that correspond to putative biological processes, and to cluster genes according to over- or under-expression in each component. We test the statistical significance of enrichment of gene annotations within clusters. ICA outperforms other leading methods, such as principal component analysis, *k*-means clustering and the Plaid model, in constructing functionally coherent clusters on microarray datasets from *Saccharomyces cerevisiae*, *Caenorhabditis elegans* and human.

Background

Microarray technology has enabled high-throughput genome-wide measurements of gene transcript levels, promising to provide insight into biological processes involved in gene regulation. To aid such discoveries, mathematical and computational tools are needed that are versatile enough to capture the underlying biology, and simple enough to be applied efficiently on large datasets.

Analysis tools fall broadly in two categories: supervised and unsupervised approaches [1]. When prior knowledge can group samples into different classes (for example, normal versus cancer tissue), supervised approaches can be used for finding gene expression patterns (features) specific to each class, and for class prediction of new samples [2-5]. Unsupervised (hypothesis-free) approaches are important for discovering novel biological mechanisms, for revealing genetic regulatory networks and for analyzing large datasets for which little prior knowledge is available. Here we apply linear and nonlinear independent component analysis (ICA) as a versatile unsupervised approach for microarray analysis, and evaluate its performance against other leading unsupervised methods.

Unsupervised analysis methods for microarray data can be divided into three categories: clustering approaches, model-based approaches and projection methods. Clustering approaches group genes and experiments with similar behavior [6-10], making the data simpler to analyze [11]. Clustering methods group genes that behave similarly under similar experimental conditions, assuming that they are are functionally related. Most clustering methods do not attempt to model the underlying biology. A disadvantage of such methods is that they partition genes and experiments into mutually exclusive clusters, whereas in reality a gene or an experiment may be part of several biological processes. Model-based approaches first generate a model that explains the interactions among biological entities participating in genetic regulatory networks, and then train the parameters of the model on expression datasets [12-16]. Depending on the complexity of the model, one challenge of model-based approaches is the lack of sufficient data to train the parameters, and another challenge is the prohibitive computational requirement of training algorithms.

Projection methods linearly decompose the dataset into components that have a desired property. There are largely two

kinds of projection methods: principal component analysis (PCA) and ICA. PCA projects the data into a new space spanned by the principal components. Each successive principal component is selected to be orthonormal to the previous ones, and to capture the maximum information that is not already present in the previous components. PCA is probably the optimal dimension-reduction technique according to the sum of squared errors [17]. Applied to expression data, PCA finds principal components, the eigenarrays, which can be used to reduce the dimension of expression data for visualization, filtering of noise and for simplifying the subsequent computational analyses [18,19].

In contrast to PCA, ICA decomposes an input dataset into components so that each component is statistically as independent from the others as possible. A common application of ICA is in blind source separation (BSS) problems [20]: suppose that there are M independent acoustic sources - such as speech, music, and others - that generate signals simultaneously, and N microphones around the sources. Each microphone records a mixture of the M independent signals. Given N mixed vectors as the signals received from the microphones, where $N \geq M$, ICA retrieves M independent components that are close approximations of the original signals up to scaling. ICA has been used successfully in BSS of neurobiological signals such as electroencephalographic (EEG) and magnetoencephalographic (MEG) signals [21-23], functional magnetic resonance imaging (fMRI) data [24] and for financial time series analysis [25,26]. ICA can also be used to reduce the effects of noise or artifacts of the signal [27] because usually noise is generated from independent sources. Most applications of ICA assume that the source signals are mixed linearly into the input signals, and algorithms for linear ICA have been developed extensively [28-32]. In several applications nonlinear mixtures may provide a more realistic model and several methods have been developed recently for performing nonlinear ICA [33-35]. Liebermeister [36] first proposed using linear ICA for microarray analysis to extract expression modes, where each mode represents a linear influence of a hidden cellular variable. However, there has been no systematic analysis of the applicability of ICA as an analysis tool in diverse datasets, or comparison of its performance with other analysis methods.

Here we apply linear and nonlinear ICA to microarray data analysis to project the samples into independent components. We cluster genes in an unsupervised fashion into non-mutually exclusive clusters, based on their load in each independent component. Each retrieved independent component is considered a putative biological process, which can be characterized by the functional annotations of genes that are predominant within the component. To perform nonlinear ICA, we applied a methodology that combines the simplifying kernel trick [37] with a generalized mixing model. We systematically evaluate the clustering performance of several ICA methods on five expression datasets, and find that overall ICA is superior to other leading clustering methods that have been used to analyze the same datasets. Among the different ICA methods, the natural-gradient maximum-likelihood estimation (NMLE) method [28,29] is best in the two largest datasets, while our nonlinear ICA method is best in the three smaller datasets.

Results

Mathematical model of gene regulation

We model the transcription level of all genes in a cell as a mixture of independent biological processes. Each process forms a vector representing levels of gene up-regulation or down-regulation; at each condition, the processes mix with different activation levels to determine the vector of observed gene expression levels measured by a microarray sample (Figure 1). Mathematically, suppose that a cell is governed by M independent biological processes $S = (s_1, ..., s_M)^T$, each of which is a vector of K gene levels, and that we measure the levels of expression of all genes in N conditions, resulting in a microarray expression matrix $X = (x_1,..., x_N)^T$. We define a model whereby the expression level at each different condition j can be expressed as linear combinations of the M biological processes: $x_j = a_{j1}s_1+...+a_{jM}s_M$. We can express this model concisely in matrix notation (Equation 1).

$$X = AS, \qquad \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{1M} \\ \vdots & & \vdots \\ a_{N1} & \cdots & a_{NM} \end{bmatrix} \begin{bmatrix} s_1 \\ \vdots \\ s_M \end{bmatrix} \qquad (1)$$

When the matrix X represents log ratios $x_{ij} = \log_2(R_{ij}/G_{ij})$ of red (experiment) and green (reference) intensities (Figure 1), Equation 1 corresponds to a multiplicative model of interactions between biological processes. More generally, we can express $X = (x_1,..., x_N)^T$ as a post-nonlinear mixture of the underlying independent processes (Equation 2, where $f(.)$ is a nonlinear mapping from N to N dimensional space).

$$X = f(AS), \qquad \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} = f\left(\begin{bmatrix} a_{11} & \cdots & a_{1M} \\ \vdots & & \vdots \\ a_{N1} & \cdots & a_{NM} \end{bmatrix} \begin{bmatrix} s_1 \\ \vdots \\ s_M \end{bmatrix} \right) \qquad (2)$$

A nonlinear mapping $f(.)$ could represent interactions among biological processes that are not necessarily linear. Examples of nonlinear interactions in gene regulatory networks include the AND function [38] or more complex logic units [39], toggle switch or oscillatory behavior [40], multiplicative effects resulting from expression cascades; for further examples see also [41].

Since we assume that the underlying biological processes are independent, we can view each of the vectors $s_1,..., s_M$ as a set of K samples of an independent random source. Then, ICA can be applied to find a matrix W that provides the transformation $Y = (y_1,..., y_M)^T = WX$ of the observed matrix X under which the transformed random variables $y_1,..., y_M$, called the

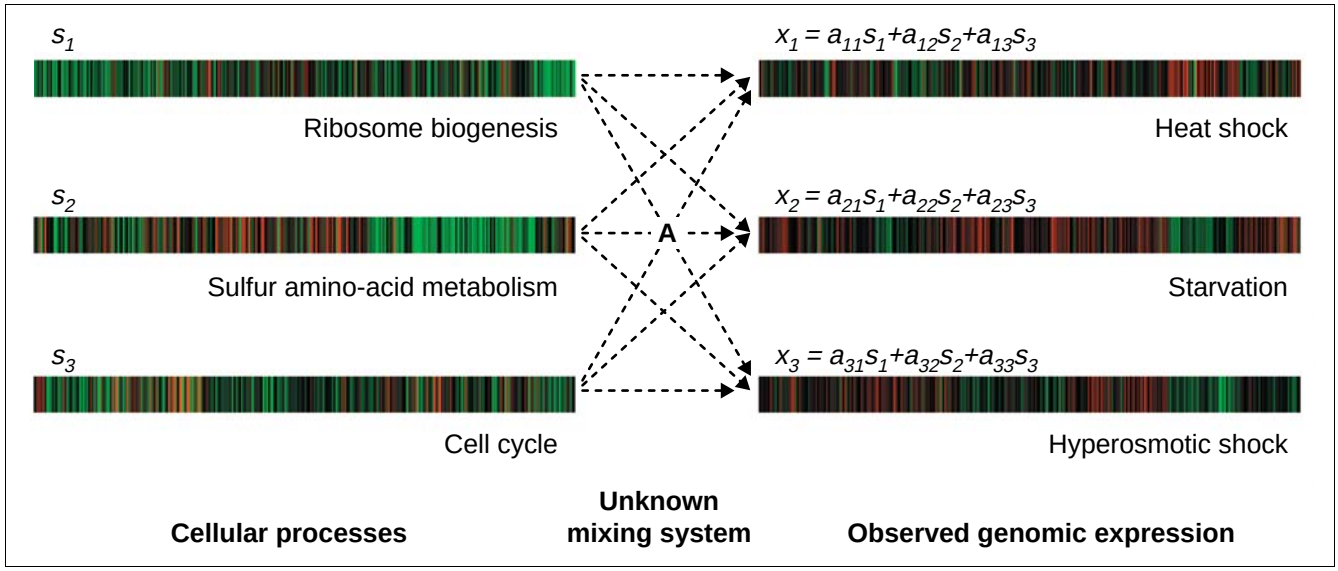

Figure 1
Model of gene expression within a cell. Each genomic expression pattern at a given condition, denoted by x_i, is modeled as linear combination of genomic expression programs of independent biological processes. The level of activity of each biological process is different in each environmental condition. The mixing matrix A contains the linear coefficients a_{ij}, where a_{ij} = activity level of process j in condition i. The example shown uses data generated by Gasch *et al.* [48].

independent components, are as independent as possible [42]. Assuming certain mathematical conditions are satisfied (see Discussion), the retrieved components $y_1,..., y_M$ are close approximations of $s_1,..., s_M$ up to permutation and scaling.

Methodology

Given a matrix X of N microarray measurements of K genes, we perform the following steps:

Step 1 - ICA-based decomposition. Use ICA to express X according to Equation 1 or 2, as a mixture of independent components $y_1, ..., y_M$. Each component y_i is a vector of K *loads* $y_i = (y_{i1}, ..., y_{iK})$ where the j^{th} load corresponds to the j^{th} gene on the original expression data.

Step 2 - clustering. Cluster the genes according to their relative loads y_{ij} in the components $y_1, ..., y_M$. A gene may belong to more than one cluster and some genes may not belong to any clusters.

Step 3 - measurement of significance. Measure the enrichment of each cluster with genes of known functional annotations.

ICA-based decomposition

Prior to applying ICA, we normalize the expression matrices X to contain log ratios $x_{ij} = \log_2(R_{ij}/G_{ij})$ of red and green intensities and we remove any samples that are closely approximated as linear combinations of other samples. We find as many independent components as samples in the input dataset, that is, $M = N$ (see Discussion). The algorithms we use for ICA are described in Methods.

Clustering

Based on our model, each component is a putative genomic expression program of an independent biological process. Our hypothesis is that genes showing relatively high or low expression levels within the component are the most important for the process. First, for each independent component, we sort genes by the loads within the component. Then we create two clusters for each component: one cluster containing C% of all genes with larger loads, and one cluster containing C% of genes with smaller loads.

$$\text{Cluster}_{i,1} = \{\text{gene j} \mid y_{ij} = (C\% \times K)^{\text{th}} \text{ largest load in } y_i\}$$

$$\text{Cluster}_{i,2} = \{\text{gene j} \mid y_{ij} = (C\% \times K)^{\text{th}} \text{ smallest load in } y_i\} \quad (3)$$

In Equation 3, y_i is the i^{th} independent component, a vector of length K; and C is an adjustable coefficient.

Measurement of biological significance

For each cluster, we measure the enrichment with genes of known functional annotations.

In our datasets we measured the biological significance of each cluster as follows. For datasets 1-4, we used the Gene Ontology (GO) [43] and the Kyoto Encyclopedia of Genes and Genomes (KEGG) [44] annotation databases. We combined all annotations in 502 gene categories for yeast, and 996

categories for *C. elegans* (see Methods). For dataset 5, we used the seven categories of tissues annotated by Hsiao *et al.* [45]. We matched each ICA cluster with every category and calculated the p value, that is, the chance probability of the observed intersection between the cluster and the category (see Methods for details). We ignored categories with p values greater than 10^{-7}. Assuming that there are at most 1,000 functional categories and roughly 500 ICA clusters, any p value larger than $1/(500 \times 1{,}000) = 2 \times 10^{-6}$ is not significant.

Evaluation of performance

Expression datasets

We applied ICA on the following five expression datasets (Table 1): dataset 1, budding yeast during cell cycle and CLB2/CLN3 overactive strain [46], consisting of spotted array measurements of 4,579 genes in 22 experimental conditions; dataset 2, budding yeast during cell cycle [47] consisting of Affymetrix oligonucleotide array measurements of 6,616 genes in synchronized cell cultures at 17 time points; dataset 3, yeast in various stressful conditions [48] consisting of spotted array measurements of 6,152 genes in 173 experimental conditions that include temperature shocks, hyper- and hypoosmotic shocks, exposure to various agents such as peroxide, menadione, diamide, dithiothreitol, amino acid starvation, nitrogen source depletion and progression into stationary phase; dataset 4, *C. elegans* in various conditions [8] consisting of spotted array measurements of 11,917 genes in 179 experimental conditions and 17,817 genes in 374 experimental conditions that include growth conditions, developmental stages and a variety of mutants; and dataset 5, normal human tissue [45] consisting of Affymetrix oligonucleotide array measurements of 7,070 genes in 59 samples of 19 kinds of tissues. We used KNNimpute [49] to fill in missing values. For each dataset, first we decomposed the expression matrix into independent components using ICA, and then we performed clustering of genes based on the decomposition.

We evaluated the performance of ICA in finding components that result in gene clusters with biologically coherent annotations, and compared our results with the performance of other methods that were used to analyze the same datasets. In particular, we compared with the following methods: PCA, which Alter *et al.* [18] applied to the analysis of the yeast cell cycle data (dataset 1) and Misra *et al.* [19] applied to the analysis of human tissue data (dataset 5); k-means clustering, which Tavazoie *et al.* [10] applied to the yeast cell cycle data (dataset 2); the Plaid model [14] applied to the dataset of yeast cells under stressful conditions (dataset 3); and the topographical map-based method (topomap) that Kim *et al.* [8] applied to the *C. elegans* data (dataset 4). In all comparisons we applied the natural-gradient maximum-likelihood estimation (NMLE) ICA algorithm [28,29] for linear ICA, and a kernel-based nonlinear BSS algorithm [34] for nonlinear ICA. The single parameter in our method was the coefficient C in Equation 3, with a default $C = 7.5\%$.

Detailed results and gene lists for all the clusters that we obtained with our methods are provided in the web supplements in [50].

Comparison of ICA with PCA

Alter *et al.* [18] introduced the use of PCA in microarray analysis. They decomposed a matrix X of N experiments × K genes into the product $X = U \Sigma V^T$ of a $N \times L$ orthogonal matrix U, a diagonal matrix Σ, and a $K \times L$ orthogonal matrix V, where $L = rank(X)$. The columns of U are called the eigengenes, and the columns of V are called the eigenarrays. Both eigenarrays and eigengenes are uncorrelated. Alter *et al.* [18] hypothesized that each eigengene represents a transcriptional regulator and the corresponding eigenarray represents the expression pattern in samples where the regulator is overactive or underactive.

ICA expresses X as a product $X = AS$ (Equations 1 and 2), where S is an $L \times K$ matrix whose rows are statistically-independent profiles of gene expression. The main mathematical difference between ICA and PCA is that PCA finds L uncorrelated expression profiles, whereas ICA finds L statistically-independent expression profiles. Statistical independence is a stronger condition than uncorrelatedness. The two

Table 1

The five datasets used in our analysis

Source (paper, datasets)	Array type	Description	Number of genes	Number of experiments
[46,52]	Spotted	Budding yeast during cell cycle and CLB2/CLN3 overactive strain	4,579	22
[47,54]	Oligonucleotide	Budding yeast during cell cycle	6,616	17
[48,56]	Spotted	Yeast in various stressful conditions	6,152	173
[8,59]	Spotted	*C. elegans* in various conditions	17,817	553
[45,53]	Oligonucleotide	Normal human tissue including 19 kinds of tissues	7,070	59

For each dataset, the source of the dataset, the type of microarray, the organism, a short description of the experimental conditions, the number of genes, and the number of experiments, are shown.

mathematical conditions are equivalent for Gaussian random variables, such as random noise, but different for non-Gaussian variables. We hypothesized that biological processes have highly non-Gaussian distributions, and therefore will be best separated by ICA. To test this hypothesis we compared ICA with PCA on datasets 1 and 5, which Alter *et al.* [18] and Misra *et al.* [19], respectively, analyzed with PCA.

Alter *et al.* [18] preprocessed dataset 1 with normalization and degenerate subspace rotation, and subsequently applied PCA to recover 22 eigengenes and 22 eigenarrays. The expression matrix they used consists of ratios $x_{ij} = (R_{ij}/G_{ij})$ between the red and green intensities. Since a logarithm transformation is the most commonly used method for variance normalization [51], we used data processed to contain log-ratios $x_{ij} = \log_2(R_{ij}/G_{ij})$ between red and green intensities obtained from [52]. We applied ICA to the microarray expression matrix X without any preprocessing, and found 22 independent components. We compared the biological coherence of 44 clusters consisting of genes with significantly high or low expression levels within the independent components, with clusters similarly obtained from the principal components of Alter *et al.* [18]. (We used the most favorable clustering coefficient C for each of the principal components and independent components, see Methods); C was fixed to 17.5 for ICA but it was varied from five to 45 with an interval of 2.5 for PCA and the result for $C = 37.5$ (best) is illustrated in Figure 2a, while three of others are illustrated in Figure 2b. For each cluster, we calculated p values with every functional category from GO and KEGG, and retained functional categories with p value < 10^{-7}. This resulted in 13 functional categories covered only with PCA clusters, 27 only with ICA clusters, and 33 with both. Categories covered by either method but not both, typically had high p values (low significance). For functional categories detected by either ICA or PCA clusters, we made a scatter plot to compare the negative log of the best p values of each category (Figure 2a). In the majority of the functional categories ICA produced significantly lower p values than PCA did. For instance, among the functional categories with p value < 10^{-7}, ICA outperformed PCA in 28 out of 33 cases, with a median difference of 7.3 in $-\log_{10}$ (p value) in the 33 cases. In Figure 2a, about a half of the functional categories (13 out of 28) represented around the diagonal or under the diagonal have close connection (parent or child) within the GO tree with another category for which ICA has much smaller p value than PCA. This means that if we look at a group of similar functional categories instead of a single category, most of the groups have considerably smaller p values with ICA than with PCA. We listed the five most significant ICA clusters based on the smallest p value of functional categories within the clusters in the web supplement [50]. Cluster 13 is driven from the seventh independent component contained 915 genes that are annotated in KEGG, of which 96 are annotated as 'ribosome'-related (out of 111 total 'ribosome'-related genes in KEGG). The same cluster is highly enriched with genes annotated in GO as 'protein biosynthesis', 'structural constituent of ribosome' and 'cytosolic ribosome'. A plausible hypothesis is that the corresponding independent component represents the expression program of a biological mechanism related to protein synthesis.

We also applied ICA to another yeast cell cycle dataset using a different synchronization method produced by Spellman *et al.* [46] and to which PCA is applied by Alter *et al.* [18]. For this dataset, ICA outperformed PCA in finding significant clusters (data shown in the web supplement [50]).

We also applied nonlinear ICA to the same dataset. First, we mapped the input data from the 22-dimensional input space to a 30-dimensional feature space (see Methods). We found 30 independent components in the feature space and produced 60 clusters from these components. We compared the biological coherence of nonlinear ICA clusters to linear ICA clusters and to PCA clusters (Figure 2c,d). Overall, nonlinear ICA performed significantly better than the other methods. The five most significant clusters are shown in the web supplement [50]. Similarly to linear ICA, the most significant nonlinear ICA cluster was enriched with genes annotated as 'protein biosynthesis', 'structural constituent of ribosome', 'cytosolic ribosome' and 'ribosome' with the smallest p value being 10^{-61} for 'ribosome' compared to the p value of 10^{-51} for the corresponding ICA cluster.

Misra *et al.* [19] applied PCA to dataset 5 of 7,070 genes in 19 kinds of human normal tissue (containing 59 microarray experiments) produced by Hsiao *et al.* [45] available at [53]. The dataset they used contains 40 experiments; 19 additional microarray experiments have been performed subsequently by Hsiao *et al.* [45]. After applying PCA and a filtering method, Misra *et al.* [19] obtained 425 genes upon which they reapplied PCA and plotted a scatter plot with loadings (expression levels) of these genes in the two most dominant principal components (eigenarrays). By visual inspection they observed three linear clusters on the resulting two-dimensional plot, enriched for liver-specific, brain-specific and muscle-specific genes, respectively (no p values were provided), as annotated by Hsiao *et al.* [45]. We removed three experiments that made the expression matrix X to be nearly singular, and applied ICA on the remaining 56 experiments, resulting in 56 independent components. We generated 112 clusters using our default clustering parameter ($C = 7.5\%$), and measured the enrichment of each of the seven tissue-specific categories annotated by Hsiao *et al.* [45] within each cluster. The three most significant independent components were enriched for liver-specific, muscle-specific and vulva-specific genes with p values of 10^{-133}, 10^{-127} and 10^{-101}, respectively. The fourth most significant cluster was brain-specific (p value = 10^{-86}). In the ICA liver cluster, 214 genes were liver-specific (out of a total of 293), as compared with the 23 liver-specific genes identified by Misra *et al.* [19]. The ICA muscle cluster of 258 genes contains 211 muscle-specific genes compared to 19 muscle-specific genes identified by Misra *et al.*

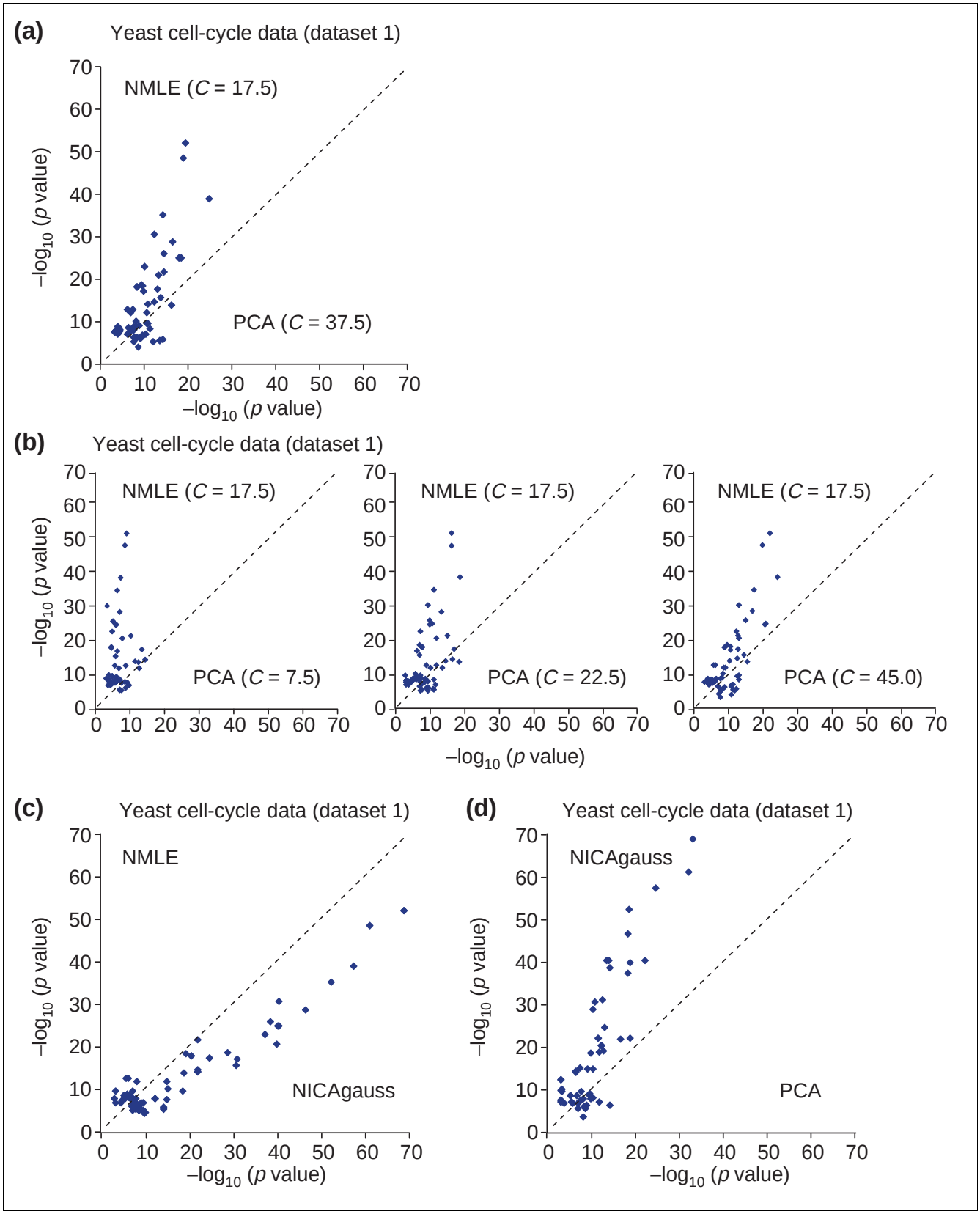

Figure 2 *(see legend on next page)*

Figure 2 *(see previous page)*
Comparison of linear ICA (NMLE), nonlinear ICA with Gaussian RBF kernel (NICAgauss), and PCA, on the yeast cell cycle spotted array data (dataset 1). For each functional category within GO and KEGG, the value of $-\log_{10}$ (*p* value) with the smallest *p* value from one method is plotted against the corresponding value from the other method. **(a)** Gene clusters based on the linear ICA components are compared with those based on PCA when *C* for PCA is fixed to its optimal value 37.5. **(b)** Gene clusters based on the linear ICA components are compared with those based on PCA with different values of *C*. **(c)** Gene clusters based on the nonlinear ICA components are compared with those based on linear ICA. **(d)** Gene clusters based on the nonlinear ICA components are compared with those based on PCA. Overall, nonlinear ICA performed slightly better than NMLE, and both methods performed significantly better than PCA.

[19]. The ICA brain cluster consisting of 277 genes contains 258 brain-specific genes compared to 19 brain-specific genes identified by Misra *et al.* [19]. We generated a three-dimensional scatter plot of the coefficients of all genes annotated by Hsiao *et al.* [45] on the three most significant ICA components (Figure 3). We observe that the liver-specific, muscle-specific and vulva-specific genes are strongly biased to lie on the x-, y- and z-axes of the plot, respectively.

We applied nonlinear ICA to this dataset (dataset 5) and the four most significant clusters from nonlinear ICA with Gaussian radial basis function (RBF) kernel were muscle-specific, liver-specific, vulva-specific and brain-specific with *p* values of 10^{-157}, 10^{-125}, 10^{-112} and 10^{-70}, respectively.

Comparison of ICA with k-means clustering

Tavazoie *et al.* [10] applied *k*-means clustering to the yeast cell cycle data generated by Cho *et al.* [47] (dataset 2) and

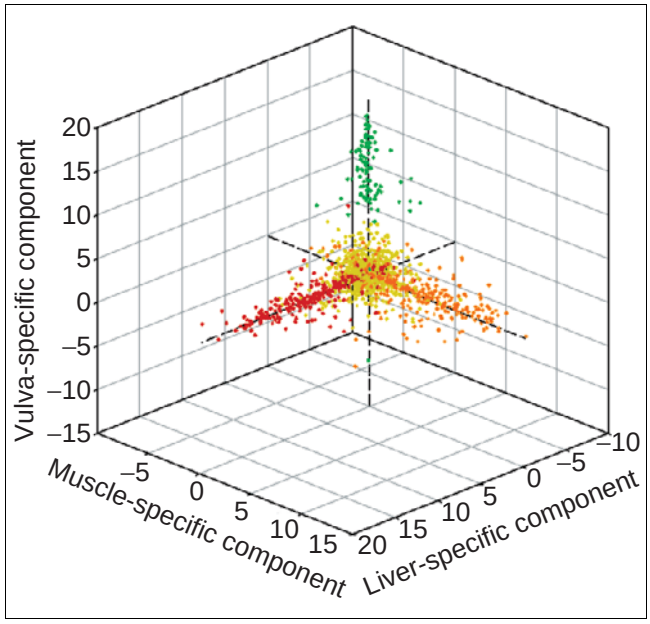

Figure 3
Three independent components of the human normal tissue data (dataset 5). Each gene is mapped to a point based on the value assigned to the gene in the 14th (*x*-axis), 15th (*y*-axis) and 55th (*z*-axis) independent components, which are enriched with liver-specific (red), muscle-specific (orange), and vulva-specific (green) genes, respectively. Genes not annotated as liver-, muscle- or vulva-specific are colored yellow.

available at [54]. First they excluded two experiments due to less efficient labeling of the mRNA during chip hybridization, and then selected 3,000 genes that exhibited the greatest variation across the 15 remaining experiments. They generated 30 clusters with *k*-means clustering, after normalizing the variance of the expression of each gene across the 15 experiments. We used the same expression dataset and normalized the variance in the same manner, but we did not remove the two problematic experiments. Instead, we removed one experiment that made the input matrix nearly singular, which destabilizes ICA algorithms. We obtained 16 independent components, and constructed 32 clusters with our default clustering parameter ($C = 7.5\%$). We collected functional categories detected with a *p* value < 10^{-7} by ICA clusters only (4), or *k*-means clusters only (16), or both (44). Categories covered by either method but not both typically had high *p* values. For functional categories detected by both ICA and *k*-means clusters, we made a scatter plot to compare the negative log of the best *p* values of the two approaches (Figure 4a). In the majority of the functional categories ICA produced significantly lower *p* values. Among the functional categories with *p* value < 10^{-7} (or 10^{-10}), ICA outperformed *k*-means clustering in 30 out of 44 (27 out of 30) cases, with a median difference of 6.1 (8.9) in $-\log_{10}$ (*p* value). The seven most significant clusters are shown in Table 2. In Figure 4a, several functional categories are represented around the diagonal. Some of them have close connections within the GO tree with other categories for which ICA has much smaller *p*-values than PCA. This means that if we look at a group of similar functional categories instead of a single category, most of the groups would have smaller *p* values with ICA than with PCA. When adjusting the parameter *C* in our method (Equation 3) from four to 14, we found similar results, with ICA still significantly outperforming *k*-means clustering (results shown in web supplement [50]).

To understand whether ICA clusters typically outperform *k*-means clusters because of larger overlaps with the GO category, or because of fewer genes outside the GO category, we defined two quantities: True Positive (TP) and Sensitivity (SN). They are determined as: $TP = k/n$ and $SN = k/f$, where *k* is the number of genes that are shared by the functional category, the cluster *n* is the number of genes within the cluster that are in any functional category and *f* is the number of genes within the functional category that appear in the microarray dataset. For all functional categories appeared in

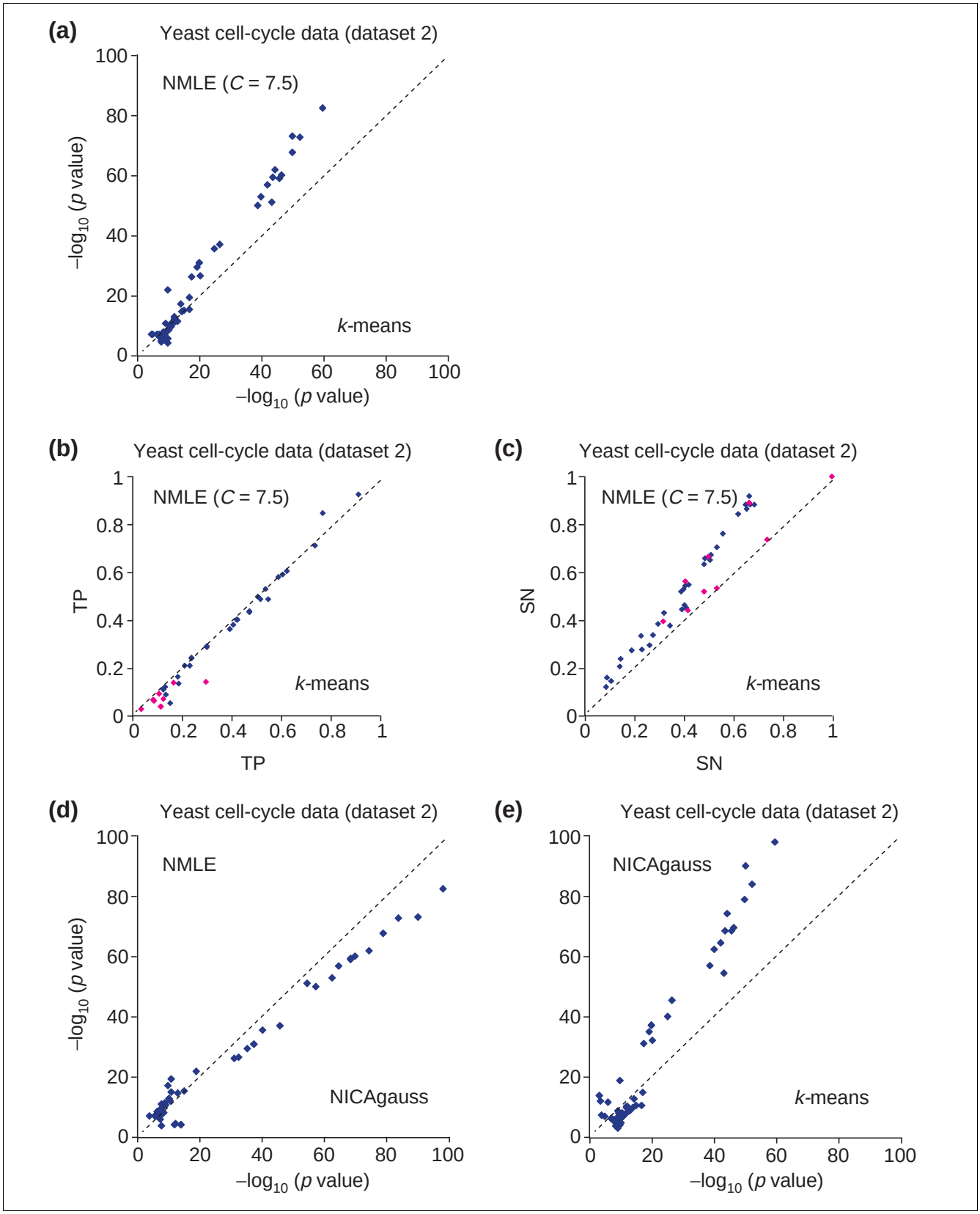

Figure 4 *(see legend on next page)*

Figure 4 *(see previous page)*
Comparison of linear ICA (NMLE), nonlinear ICA with Gaussian RBF kernel (NICAgauss), and *k*-means clustering on the yeast cell cycle oligonucleotide array data (dataset 2). For each GO and KEGG functional category, the largest $-\log_{10}(p \text{ value})$ within clusters from one method is plotted against the corresponding value from the other method. **(a)** Gene clusters based on the linear ICA components are compared with those based on *k*-means clustering. **(b)** TP (True Positives) of gene clusters based on the linear ICA components are compared with those of gene clusters based on *k*-means clustering. Functional categories for which clusters from NMLE have larger *p* values than those from *k*-means clustering algorithm are colored in purple. **(c)** SN (Sensitivity) of gene clusters based on the linear ICA components are compared with gene clusters based on *k*-means clustering. Functional categories corresponding to the ones in purple in Figure 4b are colored in purple. **(d)** Gene clusters based on the nonlinear ICA components are compared with those based on linear ICA. **(e)** Gene clusters based on the nonlinear ICA components are compared with those based on *k*-means clustering. Overall, nonlinear ICA performed better than NMLE and both methods performed better than *k*-means clustering.

Figure 4a, we compared TP and SN of ICA clusters with those of *k*-means clusters in Figure 4b and 4c, respectively. From Figure 4b and 4c, we see that ICA-based clusters usually cover more of the functional category (more sensitive), while they are comparable with *k*-means clusters in the percentage of the cluster's genes contained in the functional category (equally specific). We also applied nonlinear ICA to the same dataset. We first mapped the input data from the 16-dimensional input space to a 20-dimensional feature space (see Methods), found 20 independent components in the feature space and produced 40 clusters from these components. Comparison of the biological coherence of nonlinear ICA clusters to ICA clusters and to *k*-means clusters (Figure 4d,e) showed that overall nonlinear ICA performed significantly better than the other methods. The seven most significant nonlinear ICA clusters are shown in our web supplement [50].

Comparison of ICA with Plaid model

Lazzeroni and Owen [14] proposed the Plaid model for microarray analysis. The Plaid model takes the input expression data in the form of a matrix X_{ij} (where i ranges over N samples and j ranges over K genes). It linearly decomposes X into component matrices, namely layers, each containing nonzero values only for subsets of genes and samples in the input X that are considered to be member genes and samples of that layer. Genes that show a similar expression pattern through a set of samples, together with those samples, are assigned to be members of that layer. Each gene is assigned a load value representing the activity level of the gene in that layer. We downloaded the Plaid software from [55], and applied it to the analysis of yeast stress data of 6,152 genes in 173 experiments (dataset 3) obtained by Gasch *et al.* [48] available at [56]. We imputed dataset 3 after eliminating 868 environmental stress response (ESR) genes defined by Gasch *et al.* [48] - because clustering of the ESR genes is trivial - and obtained 173 layers. To check the biological coherence of each layer, we grouped genes showing significant activity level in each layer into clusters. For each layer, we grouped the top *C*% of up-regulated/down-regulated genes into a cluster. The value of *C* was varied from 2.5 to 42.5 with an interval of five. The setting that maximized the average *p* value of the functional categories was $C = 32.5$, with *p* value of $<10^{-20}$. (We used the most favorable clustering coefficient *C* for the Plaid model, see Methods.)

We applied ICA to the dataset (5,284 genes, 173 experiments), after we had also eliminated the 868 ESR genes that are easy to cluster. We found 173 independent components, constructed 346 clusters by using our default clustering parameters ($C = 7.5$, in Equation 3), and performed the same *p* value comparison of statistical significance with the Plaid model (Figure 5). Figure 5a compared ICA with the Plaid model when *C* is the optimal value ($C = 32.5$), and Figure 5b compared ICA with the Plaid model with *C* from 2.5 to 45. In Figure 5a, when $C = 32.5$, in the 56 functional categories detected by both the Plaid model and ICA with *p* value $<10^{-7}$, the ICA clusters had smaller *p* values for 51 out of 56 functional categories. We list the five most significant clusters from our model in Table 3. In Table 3, clusters are characterized by functional categories related to various kinds of processes for synthesis of ATP (that is, energy metabolism), whereas clusters in Table 2 are characterized by biological events occurring during the cell cycle, most of which are catabolic processes consuming ATP. This result is consistent with the fact that the many cellular stresses induce ATP depletion, which induces a drop in the ATP:AMP ratio and leads to expression of genes associated with energy metabolism [57,58]. We also applied our approach to the dataset without removing ESR genes and the results were significantly better (see our webpage at [50]).

Comparison of ICA with topomap-based clustering

Kim *et al.* [8] assembled a large and diverse dataset of 553 *C. elegans* microarray experiments produced by 30 laboratories (available at [59]). This dataset contains experiments from many different conditions, as well as several experiments on mutant worms. Of the total, 179 of the experiments contain 11,917 gene measurements, while 374 of the experiments contain 17,817 gene measurements. Kim *et al.* [8] clustered the genes with a versatile topographical map (topomap) visualization approach that they developed for analyzing this dataset. Their approach resembles two-dimensional hierarchical clustering, and is designed to work well with large collections of highly diverse microarray measurements. Using their method, they found by visual inspection 44 clusters (the mounts) that show significant biological coherence.

The ICA method is sensitive to large amounts of missing values, while methods for imputing missing values are also not appropriate in such cases. We applied ICA to the 250 experiments that had missing values for < 7,000 out of the 17,661 genes, removed four experiments that make the expression matrix to be nearly singular, and generated 492

Table 2

The seven most significant linear ICA clusters from the yeast cell cycle data (Dataset 2)

Cluster	Number of ORFs	GO/KEGG functional categories	Number of ORFs within functional category	*p* value ($\log_{10}$)
1	215	Protein biosynthesis (175)	93	-60.1
	217	Structural constituent of ribosome (118)	83	-67.6
	157	Cytosolic ribosome (94)	83	-73.1
	229	Ribosome (96)	83	-82.5
5	208	Cell cycle (220)	61	-19.5
	202	DNA-directed DNA polymerase (13)	7	-4.7
	115	Replication fork (30)	16	-9.6
	229	Cell cycle (58)	18	-6.6
11	2,072	Sulfur amino acid metabolism (12)	11	-11.1
	211	Structural constituent of cytoskeleton (25)	11	-5.9
	125	Spindle (32)	18	-10.62
9	209	Ribosome biogenesis (38)	15	-7.1
	207	RNA binding (75)	7	-3.4
	111	Nucleus (334)	54	-7.3
7	198	Glutamine family amino acid biosynthesis (11)	8	-6.8
	99	Mitochondrion (353)	22	-3.8
3	209	Protein folding (26)	11	-5.7
	212	Heat shock protein (14)	9	-6.7
11	199	DNA unwinding (10)	6	-4.5
	192	ATP-dependent DNA helicase (7)	6	-6.0
	85	Pre-replicative complex (8)	6	-5.7
	216	Cell cycle (58)	13	-3.6

The cluster IDs are shown, where cluster $C_{i,1}$ in Equation 3 is denoted by $2i-1$ and a cluster $C_{i,2}$ is denoted by $2i$. The number of genes in the cluster that have at least one annotation in GO or KEGG are listed along with the functional category with the smallest *p*-value among those in each annotation system. Four annotation systems are used: biological process (GO), molecular function (GO), cellular component (GO) and KEGG. Numbers in parentheses show the number of genes within the functional category that are present in the microarray data. Functional categories with *p*-values higher than 10^{-3} are discarded, and those with values higher than 10^{-7} are not considered to be significant. The number of genes shared by the cluster and the functional category is shown with the $\log_{10}$ of the *p*-values corresponding to each functional category for the cluster.

clusters by using our default parameters. In total, 333 GO and KEGG categories were detected by both ICA and topomap clusters with *p* values $<10^{-7}$ (Figure 6). Categories covered by either method, but not both, typically had high *p* values. We observe that the two methods perform very similarly, with most categories having roughly the same *p* value in the ICA and in the topomap clusters. The topomap clustering approach performs slightly better in a larger fraction of the categories. Still, we consider this performance a confirmation that ICA is a widely applicable method that requires minimal training, as in this case the missing values and high diversity of the data make clustering especially challenging.

We also carried out a comparison of the *TP* and *SN* quantities. For all functional categories that appeared in Figure 6a, we compared *TP* and *SN* of ICA clusters with those of topomap-driven clusters in Figure 6b and 6c, respectively. Again, typically, ICA clusters cover more genes from the functional category than the corresponding topomap clusters.

Comparison of different linear and nonlinear ICA algorithms

We tested six linear ICA methods: Natural Gradient Maximum Likelihood Estimation (NMLE) [28,29]; Joint Approximate Diagonalization of Eigenmatrices (JADE) [30]; Fast Fixed Point ICA with three decorrelation and

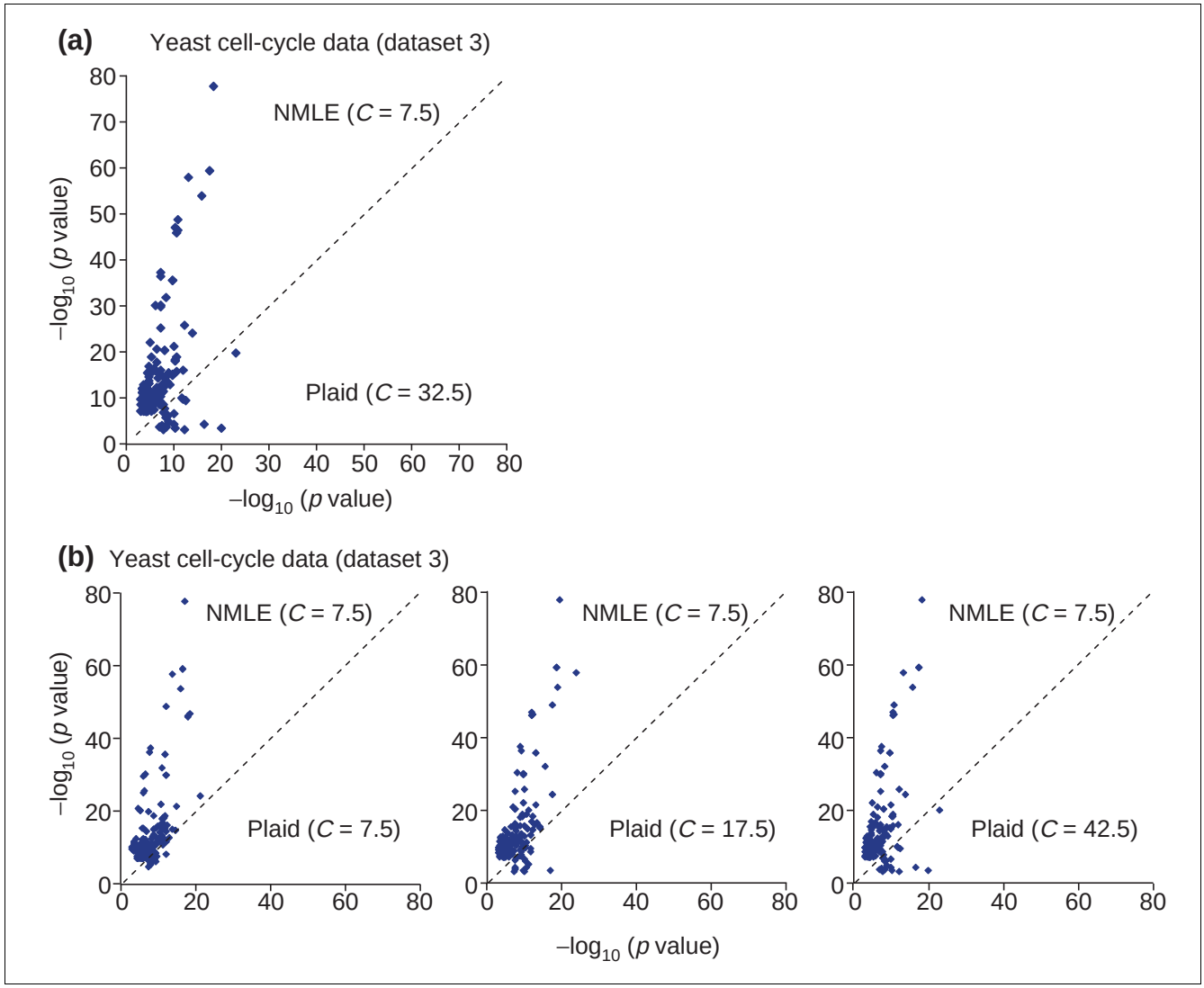

Figure 5
Comparison of linear ICA (NMLE) with the Plaid models, on the yeast stress spotted array dataset (dataset 3). For each GO and KEGG functional category, the largest -$\log_{10}$(*p* value) within clusters from one method is plotted against the corresponding value from the other method. **(a)** Gene clusters based on the NMLE components are compared with those based on the Plaid model when *C* for the Plaid model is fixed to its optimal value 32.5. **(b)** Gene clusters based on the linear ICA components are compared with those based on the Plaid model with different values of *C*.

nonlinearity approaches (different measures of non-Gaussianity: FP, FPsym and FPsymth) [31]; and Extended Information Maximization (ExtIM) [32]. We also tested two variations of nonlinear ICA: Gaussian radial basis function (RBF) kernel (NICAgauss) and polynomial kernel (NICApoly). For each dataset, we compared the biological coherence of clusters generated by each method. Among the six linear ICA algorithms, NMLE performed well in all datasets. Among both linear and nonlinear methods, the Gaussian kernel nonlinear ICA method was the best in datasets 1 and 2, the polynomial kernel nonlinear ICA method was best in dataset 5. NMLE, FPsymth and ExtM were best in the large datasets, 3 and 4. In Figure 7, we compare the NMLE method with three other ICA methods. We show the remaining comparisons in our web supplement [50]. Overall, the linear ICA algorithms consistently performed well in all datasets. The nonlinear ICA algorithms performed best in the small datasets, but were unstable in the two largest datasets.

The Extended Infomax ICA algorithm [32] can automatically determine whether the distribution of each source signal is super-Gaussian, with a sharp peak at the mean and long tails (such as the Laplace distribution), or sub-Gaussian, with a small peak at the mean and short tails (such as the uniform distribution). Interestingly, the application of Infomax ICA to all the expression datasets uncovered no source signal with sub-Gaussian distribution. A likely explanation is that the microarray expression datasets are mixtures of super-

Table 3

The six most significant linear ICA clusters from the yeast in various stress conditions data (Dataset 3)

Cluster	Number of ORFs	GO and KEGG functional category	Number of ORFs within functional category	*p*-value ($\log_{10}$)
17	378	Protein biosynthesis (181)	76	-37.3
	407	Structural constituent of ribosome (73)	63	-57.8
	225	Mitochondrion (286)	137	-77.7
	423	Translation (76)	32	-15.5
15	346	Amino acid and derivative metabolism (84)	58	-47.0
	379	Oxidoreductase (141)	39	-11.9
	423	Metabolism of other amino acids (62)	26	-12.7
1	363	TCA intermediate metabolism (19)	18	-18.9
	381	Oxidoreductase (141)	52	-22.1
	198	Mitochondrion (286)	77	-22.9
	421	Oxidative phosphorylation (137)	35	-24.2
65	377	Protein catabolism (123)	35	-11.1
	377	Threonine endopeptidase (30)	20	-15.1
	194	26S proteasome (41)	26	-18.2
	423	Proteasome (32)	21	-15.5
61	375	Main pathways of carbohydrate metabolism (51)	27	-16.5
	395	Transporter (218)	37	-4.8
	184	Cytosol (125)	32	-9.1
	421	Glycolysis/gluconeogenesis (36)	24	-17.9
75	386	Cell-cell fusion (90)	32	-12.8
	390	Transmembrane receptor (14)	6	-3.3
	189	External protective structure (66)	16	-4.3

The cluster IDs are shown where cluster $C_{i,1}$ in Equation 3 is denoted by $2i-1$ and a cluster $C_{i,2}$ is denoted by $2i$. The number of genes in the cluster that have at least one annotation in GO or KEGG are listed, along with the functional category with the smallest *p*-value among those in each annotation system. Numbers in parentheses show the number of genes within the functional category that are present in the microarray data. Functional categories with *p*-values higher than 10^{-3} are discarded, and those with values higher than 10^{-7} are not considered to be significant. The number of genes shared by the cluster and the functional category is shown with the $\log_{10}$ of the *p*-values corresponding to each functional category for the cluster.

Gaussian sources rather than of sub-Gaussian sources. This finding is consistent with the following intuition: underlying biological processes are super-Gaussian, because they affect sharply the relevant genes, typically a fraction of all genes (long tails in the distribution), and leave the majority of genes relatively unaffected (sharp peak at the mean of the distribution).

There have been several empirical comparisons of ICA algorithms using various real datasets [27,60]. Even though many of the ICA algorithms have close theoretical connections, they often reveal different independent components in real world problems. The reasons for such discrepancies are usually deviations between the assumed ICA model and the underlying behavior of real data. Discrepancies can often result from noise or wrong estimation of the source distributions. Such factors affect the convergence of each ICA algorithm differently, and therefore it is useful to apply several different ICA algorithms [27]. In our case, overall, the different ICA algorithms perform similarly. NMLE, ExtIM, and FPsymth algorithms yielded similar results except in dataset 2 where NMLE performed best. Interestingly, dataset 2 is the only one in this comparison where the data comes from oligonucleotide microarrays (Affymetrix), where the distribution is highly unbalanced and required application of variance normalization (see Methods).

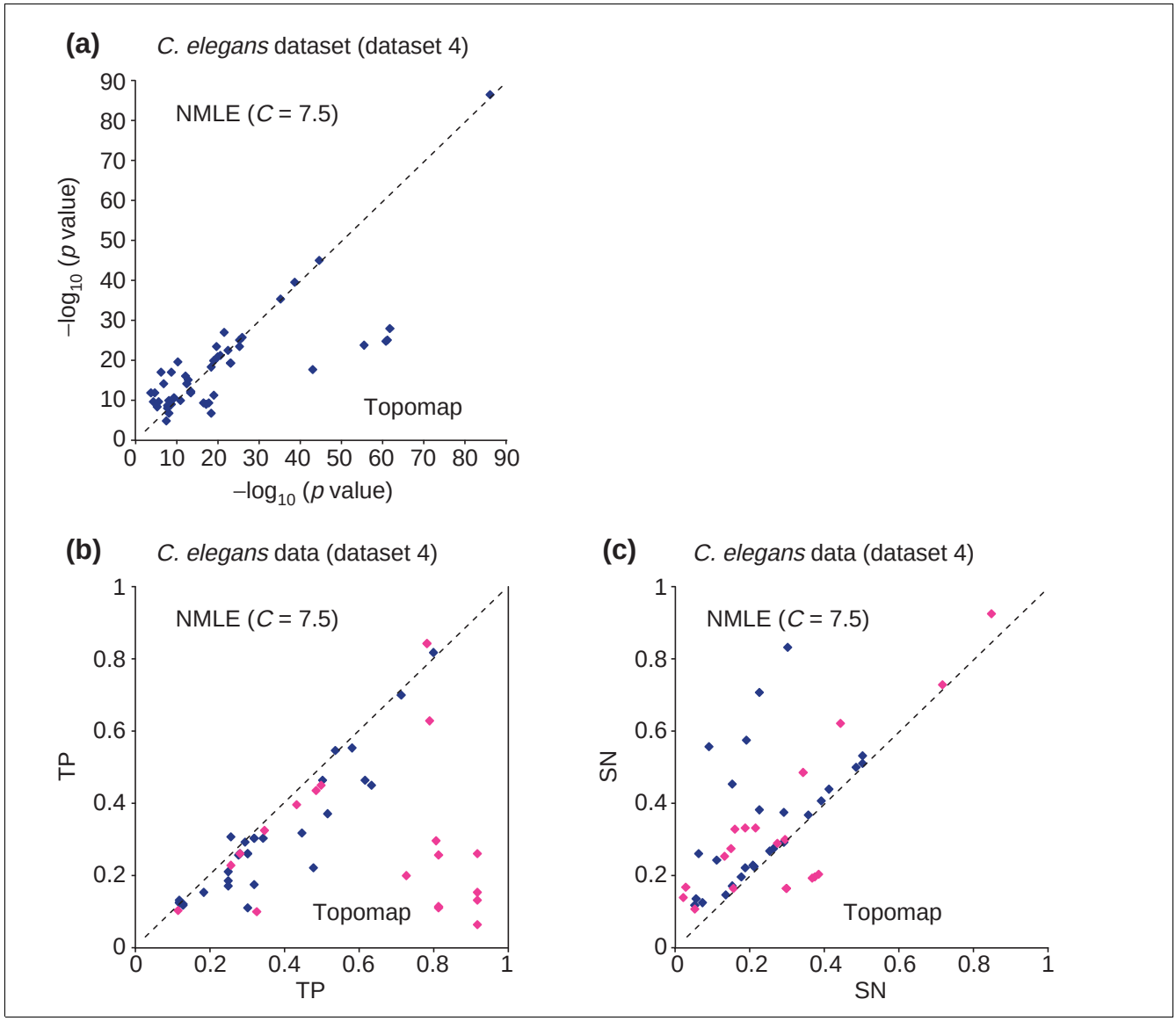

Figure 6
Comparison of linear ICA (NMLE) versus topomap-based clustering on the *C. elegans* spotted array dataset (dataset 4). For each functional category within GO and KEGG, the value of -$\log_{10}$ (*p* value) with the smallest *p* value from NMLE is plotted against the corresponding value from the topomap method. **(a)** Gene clusters based on the NMLE components are compared with those based on the Topomap method. The two methods performed comparably, as most points of low *p* values fall on the *x* = *y* axis. **(b)** TP (True Positives) of functional categories from gene clusters based on the NMLE components are compared with those of functional categories from gene clusters based on the topomap method. Functional categories for which clusters from NMLE have larger *p* values than those from topomap method are colored in purple. **(c)** SN (Sensitivity) of functional categories from gene clusters based on the linear NMLE and topomap clusters. Functional categories corresponding to the ones in purple in Figure 6b are colored in purple.

Discussion

ICA is a powerful statistical method for separating mixed independent signals. We proposed applying ICA to decompose microarray data into independent gene expression patterns of underlying biological processes and to group genes into clusters that are mutually non-exclusive with statistically significant functional coherence. Our clustering method outperformed several leading methods on a variety of datasets, with the added advantage that it requires setting only one parameter, namely the percentage ranking *C* beyond which a gene is considered to be associated with a component's cluster. We observed that performance was not very sensitive to that parameter, suggesting that ICA is robust enough to be used for clustering with little human intervention. The empirical performance of ICA in our tests supports the hypothesis that statistical independence is a good criterion for separating mixed biological signals in microarray data.

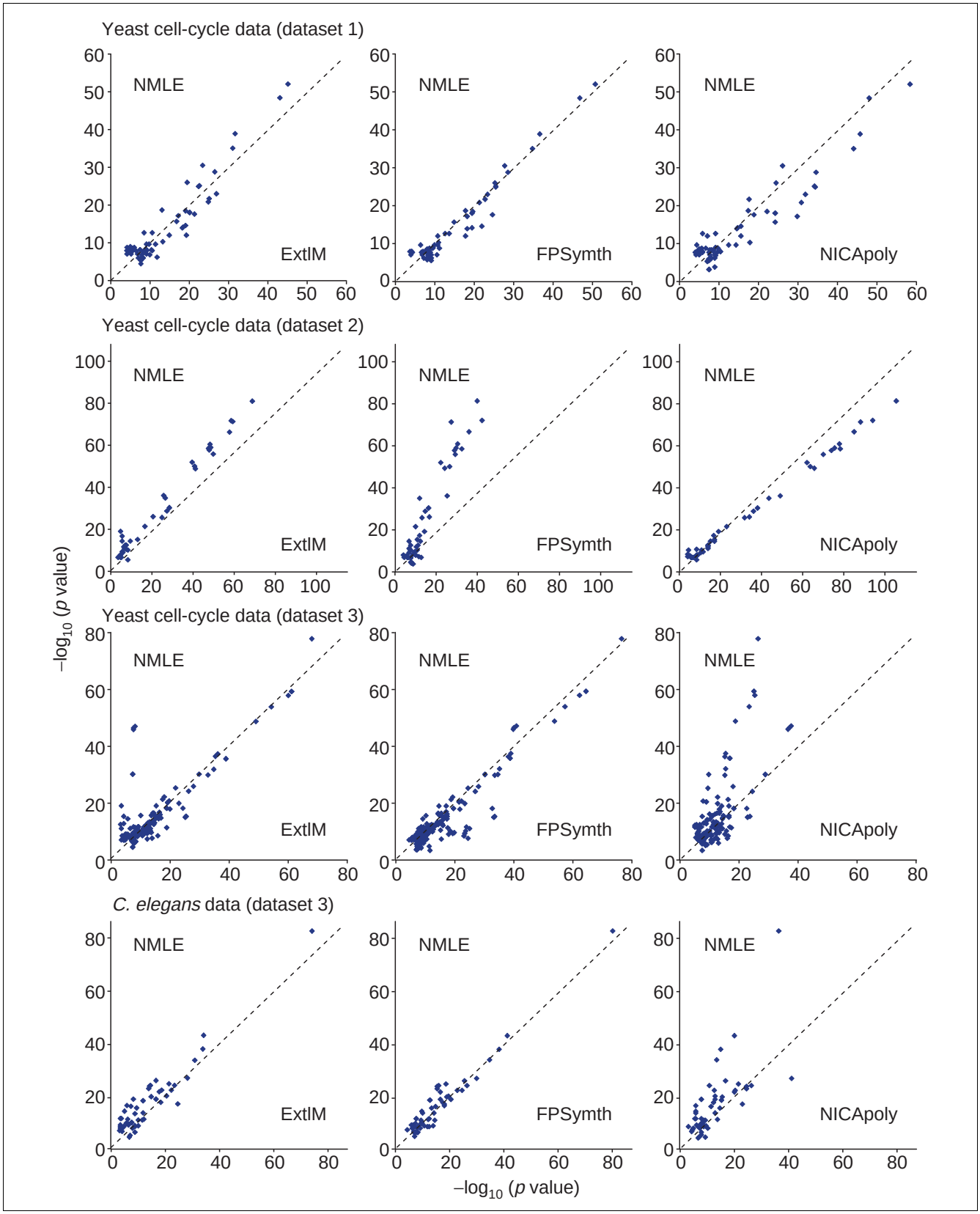

Figure 7 *(see legend on next page)*

Figure 7 *(see previous page)*
Comparison of NMLE with other ICA approaches. Comparison of the NMLE ICA algorithm with three other ICA approaches on two yeast cell cycle data (dataset 1 and 2), yeast stress data (dataset 3), and *C. elegans* data (dataset 4). Eight different ICA algorithms and variations (Table 4) were compared. The full comparison is shown in the web supplement. Overall, NMLE, ExtIM and FPsymth performed similarly except in the dataset 2. NICApoly performed comparably with NICAgauss. Both nonlinear approaches were better than NMLE in the two smaller datasets, but performed relatively poorly in the two larger datasets.

Linear ICA models a microarray expression matrix X as a linear mixture $X = AS$ of independent sources. ICA decomposition attempts to find a matrix W such that $Y = WX = WAS$ recovers the sources S (up to scaling and permutation of the components). The three main mathematical conditions for a solution to exist are [42]: the number of observed mixed signals is larger than, or equal to the number of independent sources, that is, $N = M$ in Equation 1; the columns of the mixing matrix A are linearly independent; and there is, at most, one source signal with Gaussian distribution. In microarray analysis, the first condition may mean that when too few separate microarray experiments are conducted, some of the important biological processes of the studied system may collapse into a single independent component. If the number of sources is known to be smaller than the number of observed signals, PCA is usually applied prior to ICA, to reduce the dimension of the input space. Because we expect the true number of concurrent biological processes inside a cell to be very large, we attempted to find the maximum number of independent components in our tests, which is equal to the rank of X. We also experimented with adjusting the number of independent components by randomly sampling a certain number of experiments and by using dimensional reduction using PCA for the datasets 1, 2 and 3 (results shown in the web supplement at [50]). Both random sampling and PCA dimensional reduction led to worse performance in terms of p values as the number of dimensions decreased. The main conclusion that we can draw from this drop in performance is to exclude a scenario where a small number of linearly mixing independent biological processes drove the expression of most genes in these datasets. The second condition, that the columns of the mixing matrix A are linearly independent, is easily satisfied by removing microarray experiments that can be expressed as linear combinations of other experiments, that is, those that make the matrix X singular. The third condition, that there is, at most, one source signal with Gaussian distribution, is reasonable for analyzing biological data: the most typical Gaussian source is random noise, whereas biological processes that control gene expression are expected to be highly non-Gaussian, sharply affecting a set of relevant genes, and leaving most other genes relatively unaffected. Moreover, the ability of ICA to separate a single Gaussian component may prove ideal in separating the experimental noise from expression data. This is a topic for future research.

ICA is a projection method for data analysis, but it can be interpreted also as a model-based method, where the underlying model explains the gene levels at each condition as mixtures of several statistically-independent biological processes that control gene expression. Moreover, ICA naturally leads to clustering, with each gene assigned to the clusters that correspond to independent components where the gene has a significantly high expression level. An advantage of ICA-based clustering is that each gene can be placed in zero, one or several clusters.

ICA is very similar to PCA, as both methods project a data matrix into components in a different space. However, the goals of the two methods are different. PCA finds the uncorrelated components of maximum variance, and is ideal for compressing data into a lower-dimensional space by removing the least significant components. ICA finds the statistically independent components, and is ideal for separating mixed signals. It is generally understood that ICA recovers more interesting (that is, non-Gaussian) signals than PCA does in the financial time series data [25]. If the input comprises a mixture of signals generated by independent sources, independent components are close approximates of the individual source signals; otherwise, ICA is the projection-pursuit technique that finds the projection of the high-dimensional dataset exhibiting the most interesting behavior [44]. Thus, ICA can be trusted to find statistically interesting features in the data, which may reflect underlying biological processes.

We applied a new method for performing nonlinear ICA, based on the kernel trick [37] that is usually applied in Support Vector Machine (SVM) learning [61]. Our method can deal with more general nonlinear mixture models (generalized post-nonlinear mixture models), and reduces the computation load so as to be applicable to larger datasets. Using nonlinear ICA we were able to improve performance in the three smaller datasets. However, the algorithm was still unstable in the two larger datasets. Using a Gaussian kernel, the method performed very poorly in these datasets; using a polynomial kernel, it performed comparably to linear ICA. Overall we demonstrated that nonlinear ICA is a promising method that, if applied properly, can outperform linear ICA on microarray data.

In nonlinear mixture models, the nonlinear mapping $f(.)$ represents complex nonlinear relationships between biological processes $s_1 \ldots s_M$ and gene expression data $x_{1\ldots}x_N$. In our nonlinear ICA algorithm, the nonlinear step based on kernel method is expected to map the data $x_1 \ldots x_N$ from the input space to a higher dimensional feature space where these nonlinear operations become linear so that the relationship

Table 4

Methods for performing ICA that we compared

Algorithm	Variations	Abbreviation	Description	Reference	Software
Natural Gradient Maximum Likelihood Estimation	-	NMLE	Natural gradient is applied to MLE for efficient learning	[28,29]	[72]
Extended Information Maximization	-	ExtIM	NMLE for separating mix of super- and sub-Gaussian sources	[32]	[73]
Fast Fixed-Point	Kurtosis with deflation	FP	Maximizing non-Gaussianity	[31]	[74]
	Symmetric orthogonalization	Fpsym			
	Tanh nonlinearity with symmetric orthogonalization	Fpsymth			
Joint Approximate Diagonalization of Eigenmatrices	-	JADE	Using higher-order cumulant tensor	[30]	[75]
Nonlinear ICA	Gaussian RBF kernel	NICAgauss	Kernel-based approach	[34,37,50]	[50]
	Using polynomial kernel	NICApoly			

Eight methods are based on five algorithms. The method's name, variations, abbreviation, short description, references and software that we use, are listed.

between biological processes $s_1 \ldots s_M$ and the mapped data $\Psi(x_1),\ldots, \Psi(x_N)$ becomes linear. Kernel-based nonlinear mapping has many advantages compared to other nonlinear mapping methods because it is versatile enough to cover a wide range of nonlinear operations, and at the same time reduces the computational load drastically [34]. One challenge in general is to choose the best kernel for a specific application [62,63]. A common practice is to try several different kernels, and decide on the best one empirically [3,34]. Finding which kernels best model observed nonlinear gene interactions is a direction for future research.

The linear mixture model that we proposed has the advantage of simplicity - it is expected to perform well in finding first-order features in the data, such as when a single transcription factor up-regulates a given subset of genes. Nonlinear ICA may prove capable of capturing multi-gene interactions, such as when the cooperation of several genes, or the combination of presence of some genes and absence of others, is necessary for driving the expression of another set of genes. In future research, we will attempt to capture such interactions with nonlinear modeling, and to deduce such models from the components that we obtain with nonlinear ICA. Currently our ICA model does not take into account time in experiments such as the yeast cell cycle data. A direction for future research is to incorporate a time model in our approach, whenever the microarray measurements represent successive time points.

It has been suggested that ICA be used for projection pursuit (PP) problems [64] where the goal is to find projections containing the most interesting structural information for visualization or linear clustering. ICA is applicable in this context because directions onto which the projections of the data are as non-Gaussian as possible are considered interesting [60]. Unlike the BSS problem, in the projection pursuit context inputs are not necessarily modeled as linear mixtures of independent signals. In dataset 5, we can see difference between ICA and PCA in finding interesting structures. Using a previous version of dataset 5 containing 40 out of the 59 samples, Misra *et al.* [19] constructed a scatter plot illustrating the two most dominant principal components. On that plot, there were three linearly structured clusters of genes, each turning out to be related to a tissue-specific property. Two of them were not aligned with any principal components. The corresponding plot derived by ICA in Figure 3 shows that the directions of the three most dominant independent components are exactly aligned with the linear structures of genes so that we can say that each component corresponds to a tissue sample. It is known that by applying ICA, separation can be achieved along a direction corresponding to one projection, which is not the case with PCA [60]. Nonlinear ICA can similarly be understood to be a PP method that finds nonlinear directions that are interesting in the sense of forming highly non-Gaussian distributions.

In practice, microarray expression data usually contain missing values and may have unsymmetrical distribution. In addition, there maybe false positives due to experimental errors (for example, weak signal, bad hybridization and artifact) and intrinsic error caused by dynamical fluctuations in photoelectronics, image processing, pixel-averaging, rounding error and so on. In the datasets we used in the current analysis, missing values comprise 1.427%, 3.187% and 8.55% for datasets 1, 3 and 4, respectively. The datasets have undergone log transform normalization so that equivalent fold changes in either direction have the same absolute value. Moreover, they do not have symmetrical distribution (shown in the web

supplement [50]). Missing values, unbalanced data or intrinsic errors are common challenges of many statistical techniques including ICA. Several strategies have been proposed to address these problems. In the context of microarrays and ICA, we point to the following solutions.

For missing data, a traditional way is to fill in missing values by mean imputation. However, this approach can induce some bias to the input data. The algorithm we apply, KNNimpute, is an improved version of this principle [49]. A more principled approach is to estimate density of missing data and use the estimate to fill in missing values; Expectation-Maximization (EM) algorithm is an example for this approach [65]. Chan *et al.* [65] proposed a variational Bayesian method to perform ICA on high-dimensional data containing missing values. The Bayesian ICA they proposed successfully performed with input data with missing values. New ICA algorithms with additional capability to basic ICA algorithm have been developed and it is important to develop or modify existing algorithms for use of ICA to handle wide range of gene expression datasets.

For noisy and asymmetric data, there are three potential approaches: applying data normalization and error estimation techniques; applying preprocessing step for ICA; and using advanced ICA algorithms. In the first approach, there have been approaches for normalizing spotted/oligonucleotide microarray data by estimating intrinsic errors on the data [66-68]. Applying these approaches to the dataset before ICA might help reduce errors. In the second approach, one of the successful applications of ICA is the analysis of neurobiological data such as MEG, EEG and fMRI. However, there are limitations of using ICA for these data because neurobiological data contain a lot of sensory noise and the number of independent component is unknown. Therefore, many strategies have been proposed to overcome this problem in this area. For example, Ikeda *et al.* [69] proposed a novel preprocessing step that can estimate the amount of the sensory noise and the number of sources by using factor analysis as a preprocessing step for ICA to MEG signals. In the third approach, most of the fMRI data have skewed distribution. Algorithms have been developed that support unsymmetrical source distribution [24] and those that do not require assumption on the source distribution. Stone *et al.* [24] applied skewed probability distribution for fMRI data as a way of adopting realistic physical assumption and showed improved performance of ICA. Applying these approaches to microarray analysis is an interesting future direction. Finally, a direction for future research is to use ICA as a preprocessing step, followed with subsequent analyses, such as clustering or classification methods, on the transformed space. Sophisticated clustering methods may produce more coherent groups of genes than our simple clustering scheme that groups genes with high coefficients in each component separately. Here we demonstrated that ICA transforms the data into a space whose axes have significant functional coherence, potentially making further analyses considerably more effective than when applied to the original microarray data.

Methods

Data treatment

The five datasets we used in this analysis were treated as follows.

Yeast cell cycle dataset by Spellman et al.

The yeast cell-cycle dataset in [46] was preprocessed to contain log-ratios $x_{ij} = \log_2(R_{ij}/G_{ij})$ between red and green intensities. ICA was applied on the 22 experiments with 4,579 genes that were analyzed by Alter *et al.* [18].

Yeast cell cycle dataset by Cho et al.

Variance normalization was applied to the yeast cell-cycle dataset in [47] for the 3,000 most variant genes (as in [10]). The 17th experiment, which made the expression matrix close to singular, was removed.

Yeast stress data

The yeast stress dataset in [48] was preprocessed to contain log-ratios $x_{ij} = \log_2(R_{ij}/G_{ij})$ between red and green intensities. As in Segal *et al.* [15], we eliminated 868 environmental stress response (ESR) genes defined by Gasch *et al.* [48] for which clustering is trivial and applied ICA to the remaining genes and experiments.

C. elegans *data*

Experiments in the *C. elegans* dataset [8] that contained more than 7,000 missing values were discarded. The 250 remaining experiments were used, containing expression levels for 17,817 genes preprocessed to be log-ratios $x_{ij} = \log_2(R_{ij}/G_{ij})$ between red and green intensities.

Human normal tissue

The expression levels of each gene in the human normal tissue [45] were normalized across the 59 experiments, and the logarithms of the resulting values were taken. Experiments 57, 58 and 59 were removed because they made the expression matrix nearly singular.

Gene annotation database

For the yeast and *C. elegans* datasets (datasets 1, 2, 3 and 4), we used the functional categories defined by four different annotation systems: three tree ontologies developed by the GO Consortium [43] and one from KEGG [44]. For budding yeast, we used 502 functional categories from GO and KEGG annotations: 243 functional categories from biological process (GO), 120 from molecular function (GO), 81 from cellular component (GO) and 58 from KEGG biological pathway annotation. For *C. elegans*, we used 974 functional categories: 194 categories from biological process, 458 from molecular function, 231 from cellular component, and 91 from KEGG annotations. For the human dataset (dataset 5), we

used the functional annotations compiled by Hsiao *et al.* [45]: brain (618), kidney (91), liver (279), lung (75), muscle (317), prostate (46) and vulva (103). For each cluster, we reported functional categories with p values smaller than 10^{-7}.

Calculating statistical significance

We measured the likelihood that a functional category and a cluster share the given number of genes by chance, based on the following quantities: the number of genes that are shared by the functional category and the cluster (k), the number of genes within the cluster that are in any functional category (n), the number of genes within the functional category that appear in the microarray dataset (f) and the total number of genes that appear both in the microarray dataset and in any functional category (g). Based on the hypergeometric distribution, the probability p that at least k genes are shared by the functional category and the cluster is given by Equation 4.

$$p = 1 - \sum_{i=0}^{k-1} \frac{\binom{f}{i}\binom{g-f}{n-i}}{\binom{g}{n}} \qquad (4)$$

Algorithms for ICA

The ICA problem can be formulated as

$$Y = WX \qquad (5)$$

where X represents an original expression dataset of N experiments × K genes. The goal of ICA is to find W so that components, that is, rows of Y, are statistically as independent as possible. To perform ICA, we used an ICA algorithm driven by maximum likelihood estimation (MLE), a statistical approach for finding estimations of unknown parameters that result in the highest probability for observations [27]. Applying a well-known principle of density of linear transformation to Equation 5 leads to

$$p_x(x) = |\det W| \, p_y(y) = |\det W| \prod_i p_i(w_i^T x) \qquad (6)$$

where p_x and p_y are probability density functions of the original dataset X and the components Y, respectively and p_i is the probability density function of the i^{th} row of Y. The second equality is based on the assumption that the rows of Y are independent and so p_y may be factored. The log likelihood $L(W)$ of Equation 6 is given in Equation 7.

$$L(W) = \log[p_x(x)] = \log|\det W| + \sum_{i=1}^{N} \log p_i(w_i^T x) \qquad (7)$$

Based on the gradient descent rule, a learning algorithm for finding the matrix W that maximizes the log-likelihood $L(W)$ is defined as follows.

$$\Delta W \propto \frac{\partial L(W)}{\partial W} = [W^T]^{-1} - g(Wx)x^T$$

$$\text{where, } g(y) = -\frac{\frac{\partial p(y)}{\partial y}}{p(y)} = \left[-\frac{\frac{\partial p(y_1)}{\partial y_1}}{p(y_1)}, \ldots, -\frac{\frac{\partial p(y_N)}{\partial y_N}}{p(y_N)} \right]^T \qquad (8)$$

The above learning rule was first derived by Bell and Sejnowski [29] from another approach called the Information Maximization (Infomax) approach and can be derived from negentropy maximization [70]. Amari *et al.* [28] proposed that the natural gradient method makes the above learning rule more efficient and modified the above learning rule.

$$\Delta W \propto [(W^T)^{-1} - g(Wx)x^T] \, W^T W = [I - g(y) \, y^T] W \qquad (9)$$

In the above learning rule, there is one unknown parameter, $g(y)$, a function of the probability density of the sources. In practice, it is enough to decide whether the distribution of the sources is super-Gaussian or sub-Gaussian [42]. Super-Gaussian distributions have a high peak at the mean and long tails (for example, the Laplace distribution). Sub-Gaussian distributions have a low peak at the mean and short tails (for example, the uniform distribution). In our application, since independent components are expected to be genomic expression levels of biological processes, a super-Gaussian distribution is appropriate: a biological process is likely to affect a few relevant genes strongly (long tails), and the rest weakly (high peak at the mean, usually = 0). We choose $g(y)$ to be a sigmoid function, $g(u) = 2 \tanh(u)$, when we compared various ICA algorithms.

Nonlinear ICA model

Our nonlinear ICA algorithm defines a nonlinear mixture model, called a generalized post-nonlinear mixture model, described in Equation 10.

$$X = f(AS), \qquad \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} = f\left(\begin{bmatrix} a_{11} & \cdots & a_{1M} \\ \vdots & & \vdots \\ a_{N1} & \cdots & a_{NM} \end{bmatrix} \begin{bmatrix} s_1 \\ \vdots \\ s_M \end{bmatrix} \right) \qquad (10)$$

where $f(.)$ is a nonlinear mapping from N to N dimensional space. If $f(.)$ operates componentwise, the above model is a post-nonlinear mixture model on which most research on nonlinear ICA has so far centered.

In general, an approach for nonlinear ICA of the above mixture models consists of two steps [27,71]: a nonlinear step and a linear step. A nonlinear step maps input $X = [x_1, \ldots, x_N]^T$ to another space, called a feature space. A linear step decomposes the mapped data into statistically independent components that are ideally identical to $S = [s_1, \ldots, s_M]^T$. Here, the key is to construct an appropriate feature space such that a nonlinear relationship between biological processes $s_1, \ldots, s_M$ and expression data $x_1, \ldots, x_N$ in the input space correspond to a

linear relationship between $s_1, ..., s_M$ and the mapped $x_1, ..., x_N$ in the feature space.

Harmeling *et al.* [34] proposed a technique for constructing a feature space and finding its orthonormal basis using a kernel method, called the kernel trick [37] that makes their approach computationally efficient. We adopted this approach for the nonlinear step and used NMLE for the linear step. Our approach for nonlinear ICA is as follows.

Step 1 - construct a feature space. We find orthonormal basis of a feature space using a kernel method with Gaussian RBF kernels and polynomial kernels

Step 2 - map the input data to the feature space. We map the expression data X of N experiments × K genes in the N-dimensional input space into Ψ in the L-dimensional ($L > N$) feature space.

Step 3 - apply linear ICA. We decompose the mapped data Ψ in the feature space into statistically independent components using the NMLE algorithm.

Construction of a feature space

Denoting by $x[i]$ the i^{th} column of X, the kernel method proposed by Harmeling *et al.* [34] constructs an L-dimensional feature space where a dot product of two mapped inputs $\Phi(x[i])$ and $\Phi(x[j])$ in the feature space is determined by a kernel function $k(.,.)$ in the input space (Equation 11).

$$k(x[i], x[j]) = \Phi(x[i]) \cdot \Phi(x[j]) \quad 1 \le i, j \le K \qquad (11)$$

Define L points $v_1,..., v_L$ in the input space so that their images $\Phi_V = [\Phi(v_1),...,\Phi(v_L)]$ form a basis in the feature space. Denoting by $\Phi_X = [\Phi(x[1]),...,\Phi(x[K])]$, since Φ_V is a basis of a feature space ($[\in R^L$), $span(\Phi_V) = span(\Phi_X)$ and $rank(\Phi_V) = L$ hold [34]. Then, orthonormal basis is defined as follows:

$$\Xi := \Phi_V (\Phi_V^T \Phi_V)^{-1/2} \qquad (12)$$

Here, multiplying $(\Phi_V{}^T \Phi_V)^{-1/2}$ with Φ_V leads to orthonormal matrix and it does not change the column space of Φ_V, that is, $rank(\Phi_V (\Phi_V{}^T \Phi_V)^{-1/2})$ is still L. The L points $v_1,..., v_L$ can be selected from $x[1],...x[K]$ if they fulfill $rank(\Phi_V) \approx L$, which implies that $\Phi_V{}^T \Phi_V$ is full rank [34]. To determine the best basis of the feature space, we randomly sample L input vectors from $x[1],..., x[K]$, obtain $v_1,..., v_L$, and calculate the conditional number of $\Phi_V{}^T \Phi$. We repeat this random sampling 1,000 times and choose the input vectors, $v = \{v_1, ..., v_L\}$, that results in the smallest conditional number, so that $\Phi_V{}^T \Phi_V$ becomes close to full rank. From Equation 11, we can use a kernel function $k(.,.)$ when calculating $\Phi_V{}^T \Phi_V$ (Equation 13).

$$\Phi_V^T \Phi_V = \begin{bmatrix} k(v_1,v_1) & \cdots & k(v_1,v_L) \\ \vdots & & \vdots \\ k(v_L,v_1) & \cdots & k(v_L,v_L) \end{bmatrix} \qquad (13)$$

Mapping input data to the feature space

We map all the vectors $x[1],..., x[K]$ to the feature space with Ξ as a basis. For an input vector $x[k]$, the coordinates of mapped point $\Phi(x[i])$, denoted by $\Psi(x[i])$, are calculated by a kernel function $k(.,.)$.

$$\Psi(x[i]) := \Xi^T \Phi(x[i]) = (\Phi_V^T \Phi_V)^{-1/2} \Phi_V^T \Phi(x[i]) = \begin{bmatrix} k(v_1,v_1) & \cdots & k(v_1,v_L) \\ \vdots & & \vdots \\ k(v_L,v_1) & \cdots & k(v_L,v_L) \end{bmatrix}^{-1/2} \begin{bmatrix} k(v_1,x[i]) \\ \vdots \\ k(v_K,x[i]) \end{bmatrix} \qquad (14)$$

An alternative way to find an orthonormal basis of the feature space is to use kernel principal component analysis [61]. First, calculate L eigenvectors with $\Phi_X = [\Phi(x[1]),...,\Phi(x[K])]$ such that $(\Phi_X{}^T \Phi_X/K) E_L = E_L \Lambda$ holds. Then, determine the basis in the feature space, as $\Xi: = \Phi_X E_L (K \Lambda)^{-1/2}$ and map the input vectors $x[1],..., x[K]$ into the feature space defined by these bases as:

$$\Psi(x[i]) := \Xi^T \Phi(x[i]) = \frac{1}{\sqrt{K}} \begin{bmatrix} 1/\sqrt{\lambda_1} & & 0 \\ & \ddots & \\ 0 & & 1/\sqrt{\lambda_L} \end{bmatrix} E_L \begin{bmatrix} k(x[1],x[i]) \\ \vdots \\ k(x[K],x[i]) \end{bmatrix} \qquad (15)$$

We tried this alternative method, but the results were generally worse than with random sampling.

Applying the linear ICA algorithm

We linearly decompose the mapped data $\Psi = [\Psi(x[1]),..., \Psi(x[K])] \in R^{L \times K}$ into statistically independent components using NMLE.

The requirements for a valid kernel function that specifies the feature space are described by Muller *et al.* [37]. We choose a Gaussian radial basis function (RBF) kernel described as $k(x,y) = exp(-|x - y|^2)$ and a polynomial kernel of degree 2 described as $k(x,y) = (x^T y+1)^2$. We refer to nonlinear ICA with Gaussian RBF kernel as *NICAgauss*, and with polynomial kernel as *NICApoly*.

Determination of clustering coefficient

The only adjustable parameter in our approach is the clustering coefficient C in Equation 3. When generating clusters, we varied the value from five to 15, and the result for C = 7.5% was reported. The best settings of C for each individual dataset were: 17.5% for dataset 1, 7.5% for dataset 2, 7.5% for dataset 3, 7.5% for dataset 4 and 2% for dataset 5. The best setting of C was determined to be the setting that maximizes the average of the values of - $\log_{10}$ (p-value) larger than 20. When comparing ICA with PCA and the Plaid model, C was adjusted from 5 to 45% (C = 37.5% was the optimal) and from 2.5 to 42.5% (C = 32.5% was the optimal), respectively. The favorable comparison of our approach to other methods was not sensitive to the value of C in this range.

Acknowledgements

We thank Relly Brandman, Chuong Do, Te-won Lee and Yueyi Liu for helpful edits to the manuscript. We thank Audrey Gash, and three anonymous reviewers, for helpful comments that led to a revision of our manuscript.

References

1. Butte A: **The use and analysis of microarray data.** *Nat Rev Drug Discov* 2002, **1:**951-960.
2. Ando T, Suguro M, Hanai T, Kobayashi T, Honda H, Seto M: **Fuzzy neural network applied to gene expression profiling for predicting the prognosis of diffuse large B-cell lymphoma.** *Jpn J Cancer Res* 2002, **93:**1207-1212.
3. Brown M, Grundy WN, Lin D, Cristianini N, Sugnet CW, Furey TS, Ares M, Haussler D: **Knowledge-based analysis of microarray gene expression data by using support vector machines.** *Proc Natl Acad Sci USA* 2000, **97:**262-267.
4. Golub TR, Slonim DK, Tamayo P, Huard C, Gaasenbeek M, Mesirov JP, Coller H, Loh ML, Downing JR, Caligiuri MA, *et al.*: **Molecular classification of cancer: class discovery and class prediction by gene expression monitoring.** *Science* 1999, **286:**531-537.
5. Mukherjee S, Tamayo P, Mesirov JP, Slonim D, Verri A, Poggio T: **Support vector machine classification of microarray data.** *Technical Report No. 182, AI Memo 1676* MIT, Cambridge: Massachusetts Institute of Technology; 1999.
6. Ben-Dor A, Shamir R, Yakhini Z: **Clustering gene expression patterns.** *J Comput Biol* 1999, **6:**281-297.
7. Eisen MB, Spellman PT, Brown PO, Botstein D: **Cluster analysis and display of genome-wide expression patterns.** *Proc Natl Acad Sci USA* 1998, **95:**14863-14868.
8. Kim SK, Lund J, Kiraly M, Duke K, Jiang M, Stuart JM, Eizinger A, Wylie BN, Davidson GS: **A gene expression map for *Caenorhabditis elegans*.** *Science* 2001, **293:**2087-2092.
9. Tamayo P, Slonim D, Mesirov J, Zhu Q, Kitareewan E, Dmitrovsky E, Sander ES, Golub TR: **Interpreting patterns of gene expression with self-organizing maps: method and application to hematopoietic differentiation.** *Proc Natl Acad Sci USA* 1999, **96:**2907-2912.
10. Tavazoie S, Hughes JD, Campbell MJ, Cho RJ, Church GM: **Systematic determination of genetic network architecture.** *Nat Genet* 1999, **22:**281-285.
11. Kaminski N, Friedman N: **Practical approaches to analyzing results of microarray experiments.** *Am J Respir Cell Mol Biol* 2002, **27:**125-132.
12. Bussermaker HJ, Li H, Siggia ED: **Regulatory element detection using correlation with expression.** *Nat Genet* 2001, **27:**167-174.
13. Friedman N, Linial M, Nachman I, Pe'er D: **Using Bayesian networks to analyze expression data.** *J Comput Biol* 2000, **7:**601-620.
14. Lazzeroni L, Owen A: **Plaid models for gene expression data.** *Statistica Sinica* 2002, **12:**61-86.
15. Segal E, Battle A, Koller D: **Decomposing gene expression into cellular processes.** In *Proceedings of the Eighth Pacific Symposium on Biocomputing: January 3-7 2003* Edited by: Altman RB, Durker AK, Hunter L, Jung TA, Klein TE. Kauai, Hawaii: World Scientific Publishing Company;; 2003:89-100.
16. Segal E, Barash Y, Simon I, Friedman N, Koller D: **From promoter sequence to expression.** In *Proceedings of the Sixth Annual International Conference on Research in Computational Molecular Biology: April 18-21 2002* Edited by: Myers G, Hannenhalli S, Saukoff D, Istrail S, Pevzner P, Waterman M. Washington DC: ACM Press; 2002:263-272.
17. Jolliffe IT: *Principle Component Analysis.* New York: Springer-Verlag; 1986.
18. Alter O, Brown PO, Botstein D: **Singular value decomposition for genome-wide expression data processing and modeling.** *Proc Natl Acad Sci USA* 2000, **97:**10101-10106.
19. Misra J, Schmitt W, Hwang D, Hsiao L, Gullans S, Stephanopoulos G, Stephanopoulos G: **Interactive exploration of microarray gene expression patterns in a reduced dimensional space.** *Genome Res* 2002, **12:**1112-1120.
20. Jutten C, Herault J: **Blind separation of sources, part I: an adaptive algorithm based on neuromimetic architecture.** *Signal Processing* 1991, **24:**1-10.
21. Makeig S, Bell AJ, Jung TP, Sejnowski TJ: **Independent component analysis of electroencephalographic data.** In *Proceedings of the Advances in Neural Information Processing Systems: November 27-December 2 1995; Denver, Colorado* Edited by: Touretzky D, Mozer M, Hasselmo M. Cambridge (MA): MIT Press; 1996:145-151.
22. Vigario R: **Extraction of ocular artifacts from EEG using independent component analysis.** *Electroenceph Clin Neurophysiol* 1997, **103:**395-404.
23. Vigario R, Jousmaki V, Hamalainen M, Hari R, Oja E: **Independent component analysis for identification of artifacts in magnetoencephalographic recordings.** In *Proceedings of the Advances in Neural Information Processing Systems: Dec 1-6 1997; Denver, Colorado* Edited by: Jordan M, Kearns M, Solla S. Cambridge (MA): MIT Press; 1998:229-235.
24. Stone JV, Porrill J, Porter NR, Wilkinson ID: **Spatiotemporal independent component analysis of event-related fMRI data using skewed probability density functions.** *NeuroImage* 2002, **15:**407-421.
25. Back AD, Weigend AS: **A first application of independent component analysis to extracting structure from stock returns.** *Int J Neural Syst* 1997, **8:**473-484.
26. Kiviluoto K, Oja E: **Independent component analysis for parallel financial time series.** In *Proceedings of the Fifth International Conference on Neural Information Processing: October 21-23 1998* Edited by: Kearns M, Solla S, Cohn D. Kitakyushu, Japan: MIT Press; 1999:895-898.
27. Hyvärinen A, Karhunen J, Oja E: *Independent Component Analysis.* New York: John Wiley & Sons; 2001.
28. Amari S: **Natural gradient works efficiently in learning.** *Neural Comput* 1998, **10:**251-276.
29. Bell AJ, Sejnowski TJ: **An information-maximization approach to blind separation and blind decovolution.** *Neural Comput* 1995, **7:**1129-1159.
30. Cardoso JF: **High-order contrasts for independent component analysis.** *Neural Comput* 1999, **11:**157-192.
31. Hyvärinen A: **Fast and robust fixed-point algorithms for independent component analysis.** *IEEE Transactions on Neural Networks* 1999, **10:**626-634.
32. Lee TW, Girolami M, Sejnowski TJ: **Independent component analysis using an extended Infomax algorithm for mixed subgaussian and supergaussian sources.** *Neural Comput* 1999, **11:**417-441.
33. Burel G: **Blind separation of sources: a nonlinear neural algorithm.** *Neural Networks* 1992, **5:**937-947.
34. Harmeling S, Zieche A, Kawanabe M, Muller K: **Kernel feature spaces and nonlinear blind source separation.** *Proceedings of the Advances in Neural Information Processing Systems: December 3-8 2001; Vancouver, British Columbia, Canada* Edited by: Dietterich TG, Becker S, Ghahramani Z. Cambridge (MA): MIT Press; 2002:761-768.
35. Hyvärinen A, Pajunen P: **Nonlinear independent component analysis: existence and uniqueness results.** *Neural Networks* 1999, **12:**429-439.
36. Liebermeister W: **Linear modes of gene expression determined by independent component analysis.** *Bioinformatics* 2002, **18:**51-60.
37. Muller KR, Mika S, Ratsch G, Tsudat K, Scholkopf B: **An introduction to kernel-based learning algorithms.** *IEEE Transactions on Neural Networks* 2001, **12:**181-201.
38. *The Lactose Operon* Edited by: Beckwith JR, Zipser E. New York: Cold Spring Harbor Laboratory, Cold Spring Harbor; 1970.
39. Yuh CH, Bolouri H, Davidson EH: **Genomic *cis*-regulatory logic: experimental and computational analysis of a sea urchin gene.** *Science* 1998, **279:**1896-1902.
40. Atkinson MR, Savageau MA, Myers JT, Ninfa AJ: **Development of genetic circuitry exhibiting toggle switch or oscillatory behavior in *Escherichia coli*.** *Cell* 2003, **113:**597-607.
41. Savageau MA: **Design principles for elementary gene circuits: elements, methods, and examples.** *Chaos* 2001, **11:**142-159.
42. Hyvärinen A: **Survey on independent component analysis.** *Neural Computing Surveys* 1999, **2:**94-128.
43. The Gene Ontology Consortium: **Creating the gene ontology resource: design and implementation.** *Genome Res* 2001, **11:**1425-1433.
44. Kanehisa M, Goto S: **KEGG for computational genomics.** In *Current Topics in Computational Molecular Biology* Edited by: Jiang T, Xu Y, Zhang MQ. Cambridge (MA): MIT Press; 2002:301-315.
45. Hsiao L, Dangond F, Yoshida T, Hong R, Jensen RV, Misra J, Dilon W, Lee K, Clark K, Harverty P, *et al.*: **A Compendium of gene expression in normal human tissues reveals tissue-specific genes and distinct expression patterns of housekeeping genes.** *Physiol Genomics* 2001, **7:**97-104.
46. Spellman PT, Sherlock G, Zhang MQ, Iyer VR, Anders K, Eisen MB, Brown PO, Botstein D, Futcher B: **Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization.** *Mol Biol Cell* 1998, **9:**3273-3297.
47. Cho RJ, Campbell MJ, Winzeler EA, Steinmetz L, Conway A, Wodicka L, Wolfsberg TG, Gabrielian AE, Landsman D, Lockhart DJ, *et al.*: **A genome-wide transcriptional analysis of the mitotic cell**

cycle. *Mol Cell* 1998, **2:**65-73.
48. Gasch AP, Spellman PT, Kao CM, Carmel-Harel O, Eisen MB, Storz G, Botstein D, Brown PO: **Genomic expression programs in the response of yeast cells to environmental changes.** *Mol Biol Cell* 2000, **11:**4241-4257.
49. Troyanskaya O, Cantor M, Sherlock G, Brown PO, Hastie T, Tibshirani R, Botstein D, Altman RB: **Missing value estimation methods for DNA microarrays.** *Bioinformatics* 2001, **17:**520-525.
50. **Additional data files for "Application of independent component analysis to microarrays"** [http://www.stanford.edu/~silee/ICA/]
51. Quackenbush J: **Microarray data normalization and transformation.** *Nat Genet* 2002, **32:**496-501.
52. **Yeast cell cycle analysis project** [http://cellcycle-www.stanford.edu]
53. **HuGE Index (Human Gene Expression Index)** [http://www.hugeindex.org]
54. **Additional data files for "Systematic determination of genetic network architecture"** [http://arep.med.harvard.edu/network_discovery/]
55. **Plaid models, for microarrays and DNA expression** [http://www-stat.stanford.edu/~owen/plaid/]
56. **Web supplement to "Genomic expression programs in the response of yeast cells to environmental changes"** [http://www-genome.stanford.edu/yeast_stress]
57. Hardie DG, Carling D: **The AMP-activated protein kinase: fuel gauge of the mammalian cell?** *Eur J Biochem* 1997, **246:**259-273.
58. Hardie DG: **Roles of the AMP-activated/SNF1 protein kinase family in the response to cellular stress.** *Biochem Soc Symp* 1999, **64:**13-27.
59. **Supplemental Data for "A gene expression map for *C. elegans*"** [http://cmgm.stanford.edu/~kimlab/topomap]
60. Giannakopoulos X, Karhunen J, Oja E: **An experimental comparison of neural algorithms for independent component analysis and blind separation.** *Int J Neural Syst* 1999, **9:**99-114.
61. Scholkopf B, Burges CJC, Smola AJ: *Advances in Kernel Methods - Support Vector Learning.* Cambridge (MA): MIT press; 1999.
62. Amari S, Wu S: **Improving support vector machine classifiers by modifying kernel functions.** *Neural Networks* 1999, **12:**783-789.
63. Cristianini N, Campbell C, Shawe-Taylor J: **Dynamically adapting kernels in support vector machines.** In *Proceedings of the Advances in Neural Information Processing Systems: December 1-3 1998; Denver, Colorado* Edited by: Kearns M, Solla S, Cohn D. Cambridge (MA): MIT Press; 1999:204-210.
64. Karhunen J, Oja E, Wang L, Vigario R, Joutsensalo J: **A class of neural networks for independent component analysis.** *IEEE Trans On Neural Networks* 1997, **8:**486-504.
65. Chan K, Lee TW, Sejnowsji T: **Handling missing data with variational Bayesian learning of ICA.** In *Proceedings of the Advances in Neural Information Processing Systems: 10-12 Dec 2002; Vancouver, British Columbia, Canada* Cambridge (MA): MIT Press; 2003 in press.
66. Chen Y, Dougherty ER, Bittner ML: **Ratio-based decision and the quantitative analysis of cDNA microarray images.** *J Biomed Opt* 1997, **2:**364-374.
67. Durbin BP, Hardin JS, Hawkins DM, Rocke DM: **A variance-stabilizing transformation for gene-expression microarray data.** *Bioinformatics* 2002, **18 Suppl 1:**S105-S110.
68. Rocke D, Durbin B: **A model for measurement error for gene expression arrays.** *J Comput Biol* 2001, **8:**557-569.
69. Ikeda S, Toyama K: **Independent component analysis for noisy data - MEG data analysis.** *Neural Networks* 2000, **13:**1063-1074.
70. Girolami M, Fyfe C: **Generalized independent component analysis through unsupervised learning with emergent bussgang properties.** In *Proceedings of the IEEE International Conference on Neural Networks; June 9-12 1997* Edited by: Karaayannis NB. Houston: IEEE Press; 1997:1788-1791.
71. Taleb A, Jutten C: **Source separation in post-nonlinear mixtures.** *IEEE Transaction on Signal Processing* 1999, **47:**2807-2820.
72. **Download the EEGLAB Toolbox for Matlab** [http://www.sccn.ucsd.edu/~scott/ica-download-form.html]
73. **Download Extended InfoMax for Matlab (4.2c or 5.0)** [http://www.cnl.salk.edu/~tewon/ICA/Code/ext_ica_download.html]
74. **The FASTICA Package for MatLab** [http://www.cis.hut.fi/projects/ica/fastica]
75. **Blind Source Separation and Independent Component Analysis** [http://www.tsi.enst.fr/~cardoso/guidesepsou.html]